Three_amigos

> 📎 **Three Amigos!** – название заурядного вестерна-комедии 1986 года со Стивом Мартином, Чеви Чейзом и Мартином Шортом в главных ролях. Что общего у этого фильма и разработки по методологии Agile, управляемой историями? Там есть тройка ловких и расчетливых героев, что очень здорово для семинаров по историям.

3 Амиго — это...

3 Амиго — это практика, которая дает всеобщее понимание того, что будет доставлено клиенту. Помогает доносить голос команды, а не переплетение разрозненных мнений.

Данный способ коммуникации внутри команды помогает:

- прийти к общему соглашению относительно ожиданий до начала разработки
- сформировать соглашение о том, как сделать вещь сразу правильную

А также, **способствует пониманию:**

- какую проблему мы решаем
- какие есть варианты решения
- что нам нужно сделать, чтобы задача была готова

Встреча трёх амиго — это способ совместного мышления, которое устраняет пробелы в понимании бизнес-спецификаций. Она помогает в разработке крутых пользовательских историй.

Цель заключается в том, чтобы делать работу в срок, но при этом планировать не настолько заранее, чтобы детали успевали устаревать.

Какой результат на выходе встречи 3 Амиго?

- сценарии/примеры (очевидные ответы);
- уточненные правила/критерии приемки;
- новые/декомпозированные истории;
- общее понимание проблемной области;
- эмпатия (сопереживание, участие).

Основным результатом трех амиго являются приемочные тесты, написанные в формате « Дано / Когда / Тогда». На самом деле, их написание может занять больше времени, чем хотелось бы, поэтому формализовывать все требования на встрече, когда все вместе находятся, не рекомендуется. Обычно разработчик или тестировщик работают над этим вне встречи. И как только у них все критерии и тест-кейсы записаны, 3 Амиго кратко просматривают, что получилось, чтобы убедиться, что все согласны с тем, что было записано.

Почему 3? Кем применяется?

Название практики не ограничивает нас тремя контекстами, а лишь создаёт минимальные рамки. Чтобы убедиться в том, что в проработке требований учли все технические нюансы, явные и неявные случаи, что спецификация отражает действительную нужду клиента — требуется 3 различных мышления/контекста: бизнеса, разработчика, тестировщика. При этом встреча не ограничивается только этими людьми. В ней участвуют все, кто вовлечены в реализацию требования.

- **Бизнес-аналитик или Product Owner**

  Представитель бизнеса знает (почти всегда), что он хочет получить в итоге и какое value от этого получит клиент и бизнес. Важно рассказывать об этом команде. Участвуя во встрече 3 Амиго, бизнес-аналитик делится информацией с участниками, чтобы у всех в команде было одинаковое понимание и ожидание от пользовательской истории. Также только он может ограничить scope приемочных критериев, по которым потом будет происходить приемка.

- **Разработчики**

  Разработчик знает, как реализовать требование от бизнеса, какие есть для этого возможности. Как правило, он думает о деталях, которые ему нужно знать, чтобы приступить к реализации. Задавая вопросы, исходя из своего опыта и знания системы, разработчик помогает вскрывать различные нюансы еще на этапе обсуждения требований.

- **Тестировщик**

  Тестировщик, так же, как и другие члены команды, помогает обогащать требования различными тестовыми случаями. Исходя из своего опыта, он больше и чаще подвергает сомнению любые утверждения, которые озвучивает команда. Поэтому лучше находит крайние случаи, неявные сценарии, задается вопросом, что может пойти не так, чего следует остерегаться.

Acceptance criteria

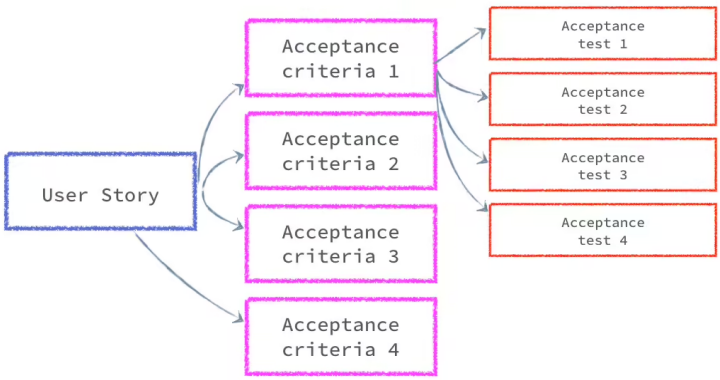
Критерии приёмки (*acceptance criteria*) — это требования от заказчика, спецификация по которой может быть проверена система/user story.

У них есть альтернативное название, ***conditions of satisfaction*** — условия удовлетворения ожиданий (автор Майк Кон).

На встречу 3-х Амиго команда уже приходит подготовленная. У них уже имеется провалидированная user story, которая, возможно, уже даже обладает каким-то набором критериев приемки, которые представитель бизнеса сформулировал самостоятельно.

Во время встречи задачи участников — дополнить/обогатить историю достаточным количеством критериев приемки для ее последующей реализации.

В критериях приемки не должны находиться детали реализации. Фактически, критерии приемки — это бизнес-правила, которым подчиняется прорабатываемая пользовательская история. Детали же реализации фиксируются в приемочных тестах, но после того, как все критерии приемки были сформулированы.



User Story

Пользовательская история (*User Story*) — короткая формулировка намерения пользователя и того, что продукт должен сделать для не

Для чего применяется User Story?

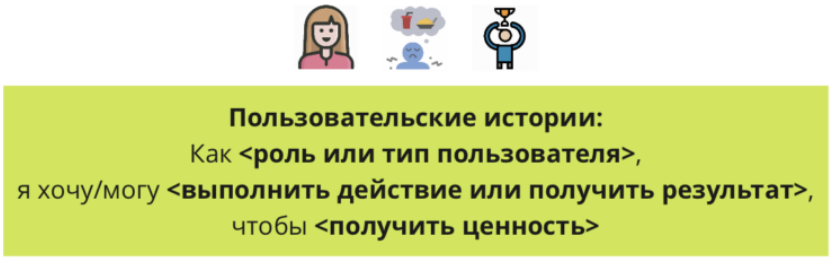
- Для описания элементов бэклога
- Для лучшего понимания пользователей

- Для описания требований к продукту на понятном для всех языке: пользователей, разработчиков другие заинтересованных лиц
- Для вовлечения в процесс разработки пользователей и заинтересованных лиц
- Для построения User Story Mapping

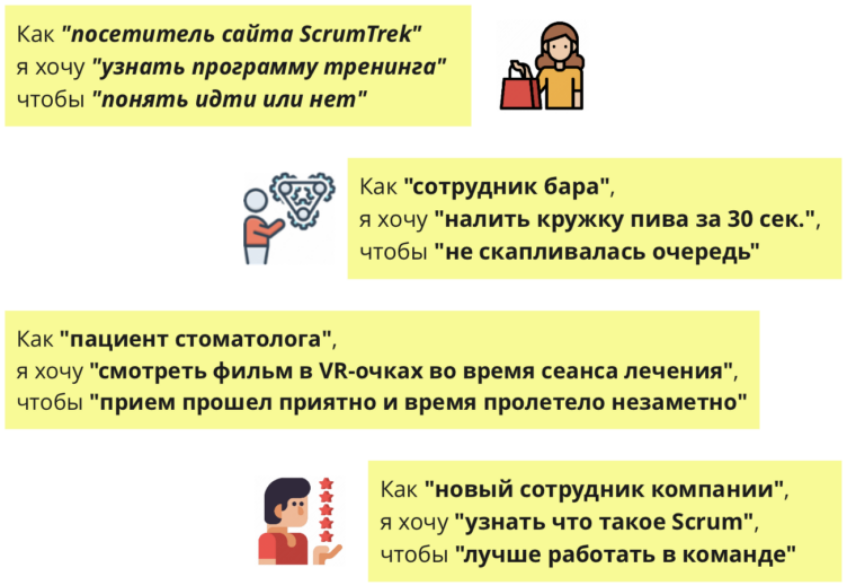
Как формулировать User Story?

User Story — это ответы на 3 вопроса, связанные в одно предложение:

- Что это за пользователь?
- Какое действие он хочет выполнить в продукте или какой результат от продукта хочет получить?
- Зачем это ему?



Примеры пользовательских историй:



Хорошая пользовательская история

INVEST — критерий хорошей истории:

Independent — независимая от других историй, то есть истории могут быть реализованы в любом порядке

Negotiable — обсуждаемая, отражает суть, а не детали; не содержит конкретных шагов реализации

Valuable — ценная для клиентов, бизнеса и стейкхолдеров

Estimable — оцениваемая по сложности и трудозатратам

Small — компактная, может быть сделана командой за одну итерацию

Testable — тестируемая, имеет критерии приемки

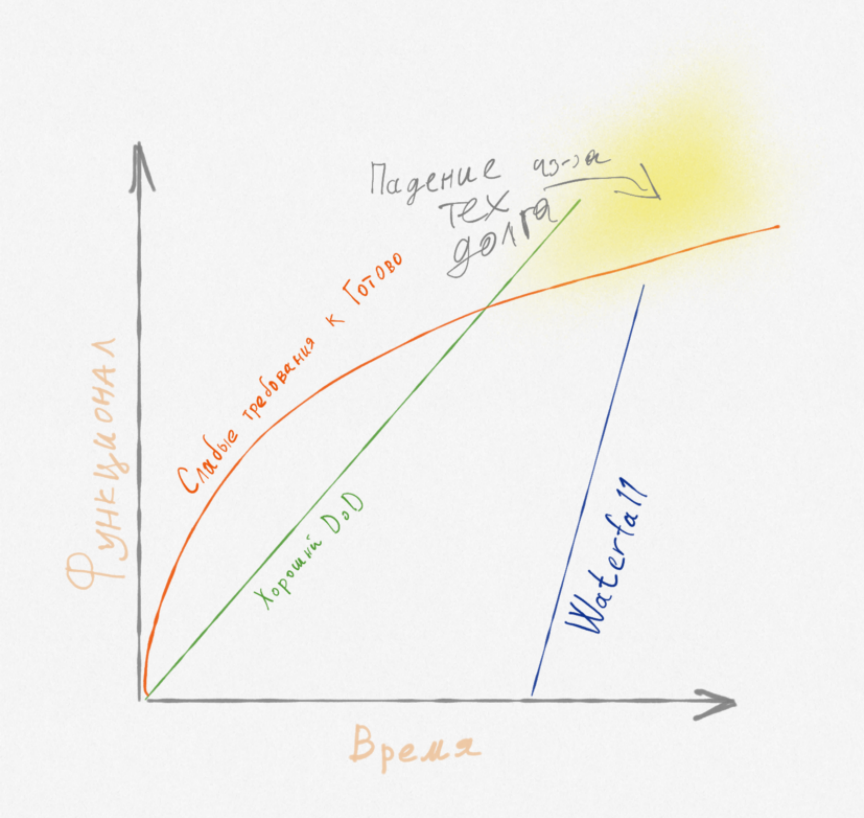
Эти критерии не всегда достижимы, но чем больше историй будут им удовлетворять, тем более гибким будет ваш процесс разработки продукта.

Definition of Done (DoD)

Критерии Готовности *(Definition of Done)* — это когда все условия или Acceptance Criteria, которым должен соответствовать программный продукт, выполнены («Done») и готовы к принятию пользователем, клиентом, командой или потребляющей системой.

Зачем нужны Definition of Done?

- Все участники **одинаково** понимали слово — Готово
- Управление качеством инкремента продукта и нефункциональными требованиями
- Взятие под контроль технического долга
- Возможность не снижать, а увеличивать скорость разработки с течением времени



Влияние DoD на скорость проекта с течением времени

Как писать Definition of Done?

Строго говоря, никаких жестких правил нет. Однако, лучше чтобы ваши DoD выглядели как чеклист. В таком случае человек либо поставит "галочку" над конкретным условием, либо — нет. Пока не поставит, пользовательскую историю нельзя считать выполненной.

Хорошие практики для больших компаний:

- Как и везде — введение чего-нибудь дополнительного — реакция на волне конкретный класс проблем. **Нет проблемы — не надо придумывать сущности.**
- При работе нескольких команд над одним продуктом необходимо использовать единый DoD.
- Включение глобальных нефункциональных требований в DoD.